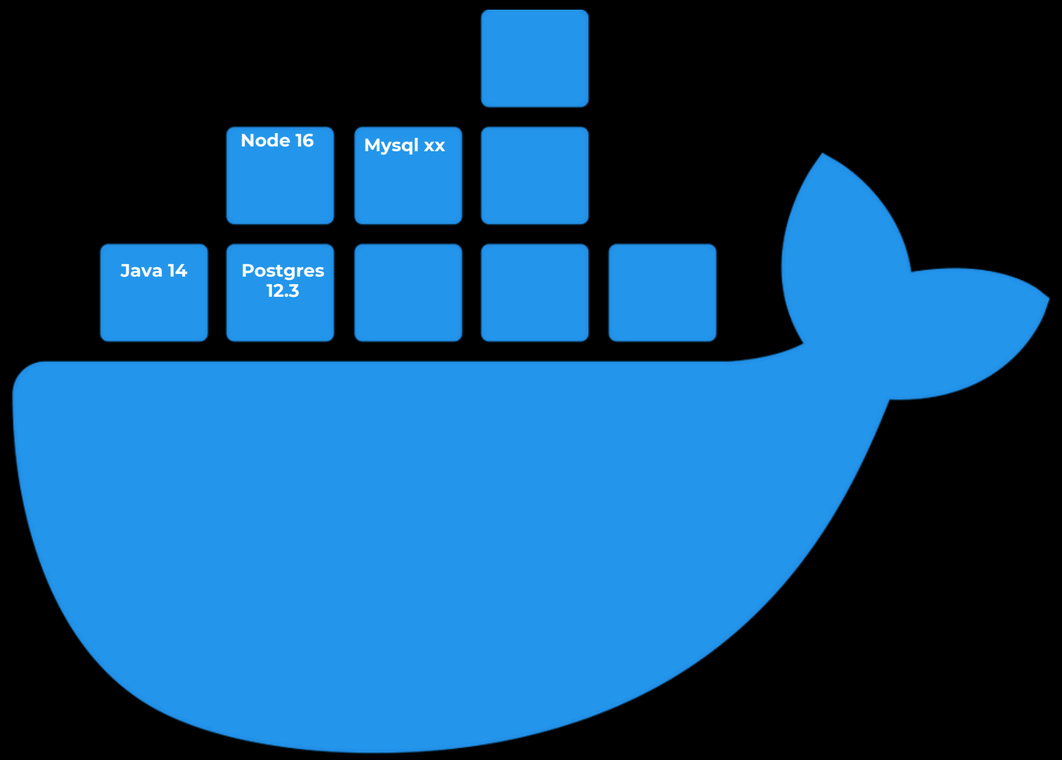
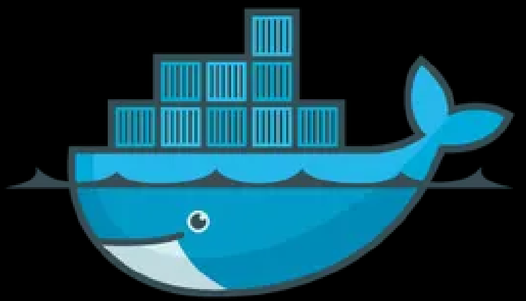


docker®

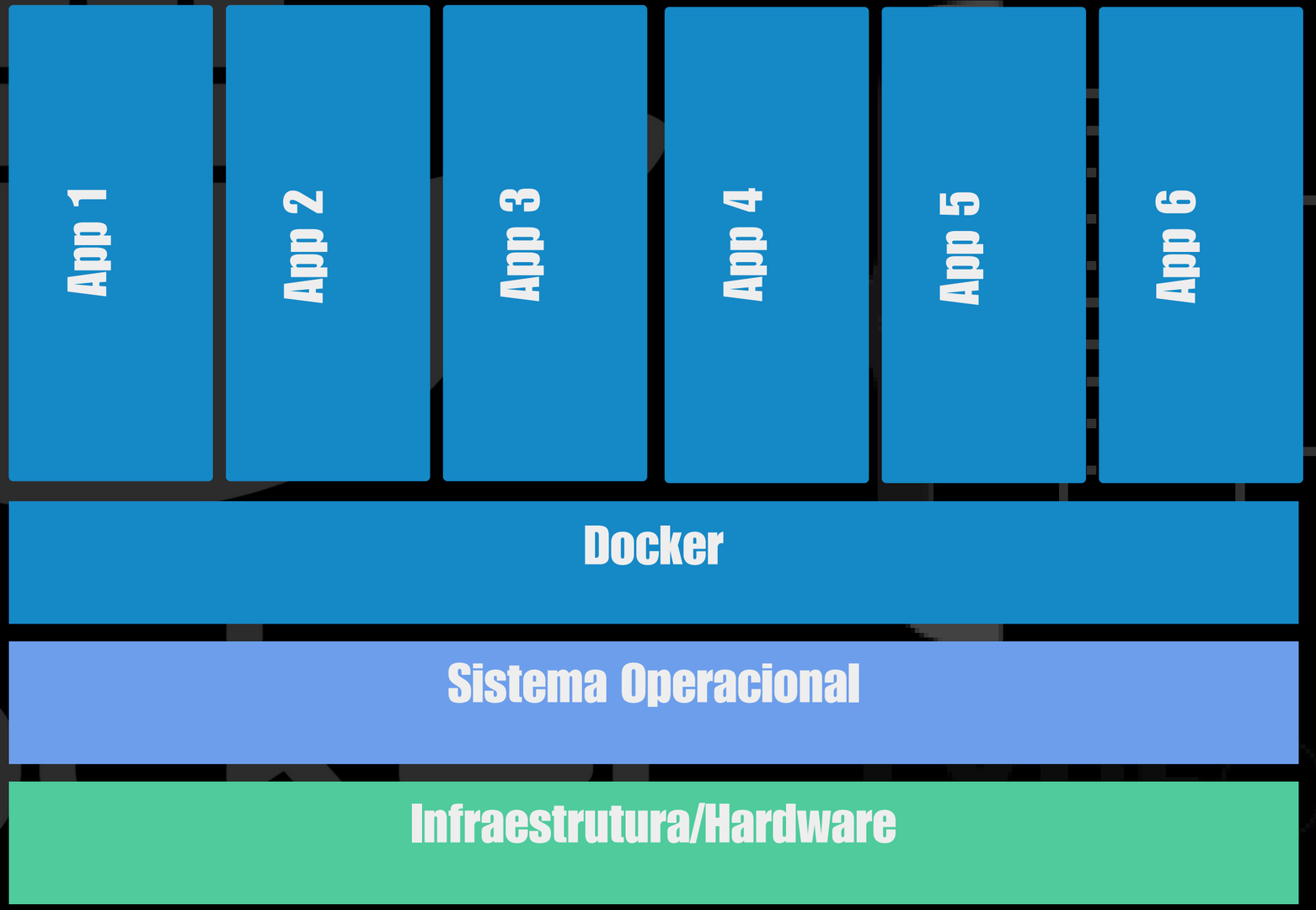


docker

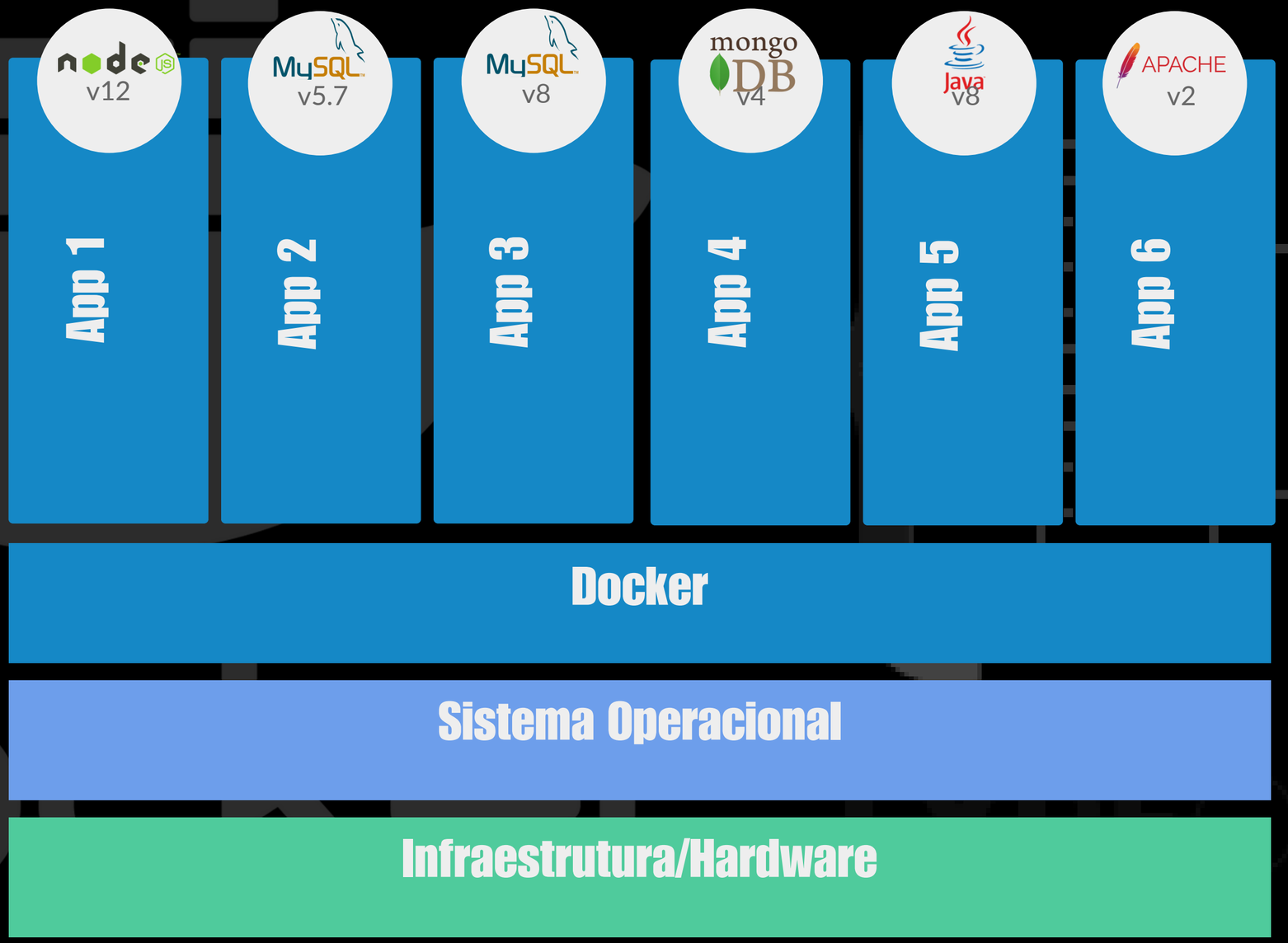
Docker

Plataforma open source que facilita a criação e gerenciamento de ambientes isolados em containers

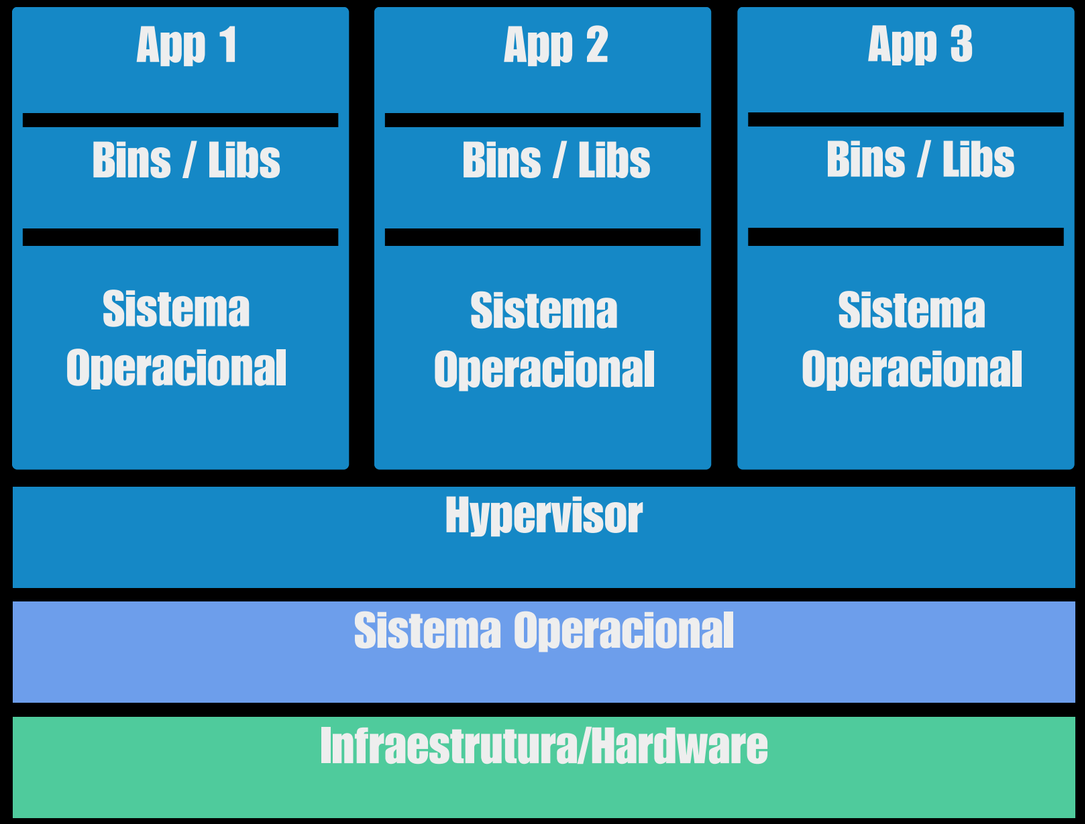
APLICAÇÕES E SERVIÇOS EM CONTAINERS



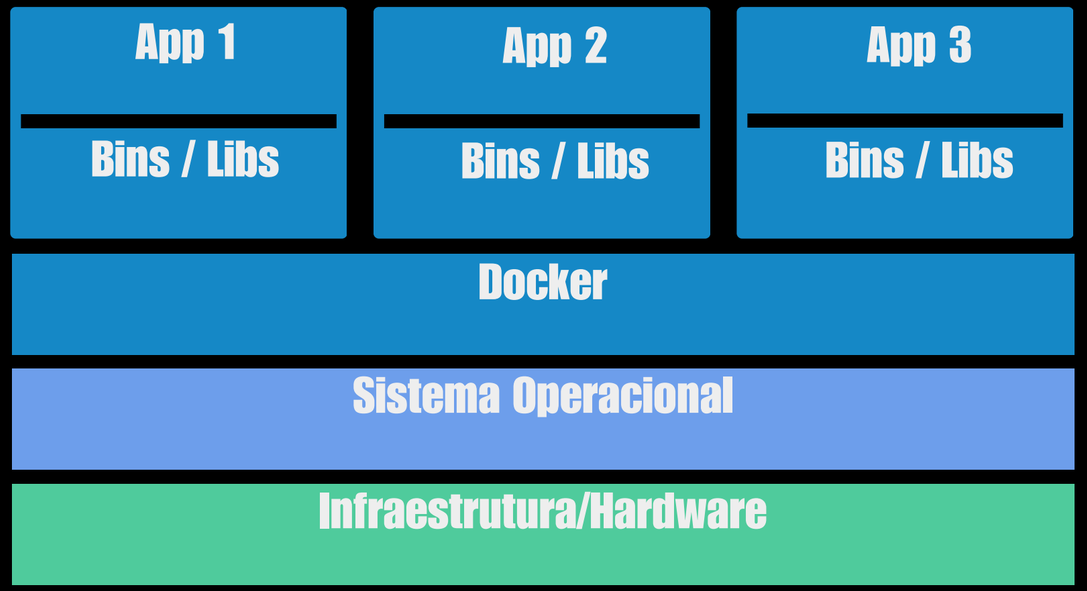
APLICAÇÕES E SERVIÇOS EM CONTAINERS



Máquinas Virtuais
x
Docker



Máquinas Virtuais



Docker Containers

Em desenvolvimento

Pontos positivos

- Subir serviços e derrubá-los sem a necessidade de instalações.
- Rodar diferentes versões de um serviço ou dependência sem esforço.
- Ambiente de desenvolvimento idêntico ao de produção.
- Maior controle do uso de recursos das aplicações.
- Tudo se torna descartável. Sem estresse na hora de testar e remover aplicações.

Em Produção

Pontos positivos

- Somente o docker instalado no host, sem mais instalação de dependências ou serviços.
- Assim como funciona em desenvolvimento, funciona em produção.
- Containers são escaláveis, replicáveis.
- Mais fácil de monitorar que aplicações fora de container.
- Em caso de falha, reiniciar aplicações e/ou serviços leva segundos.
- Fácil implementação para load balance e redundância com clusters.

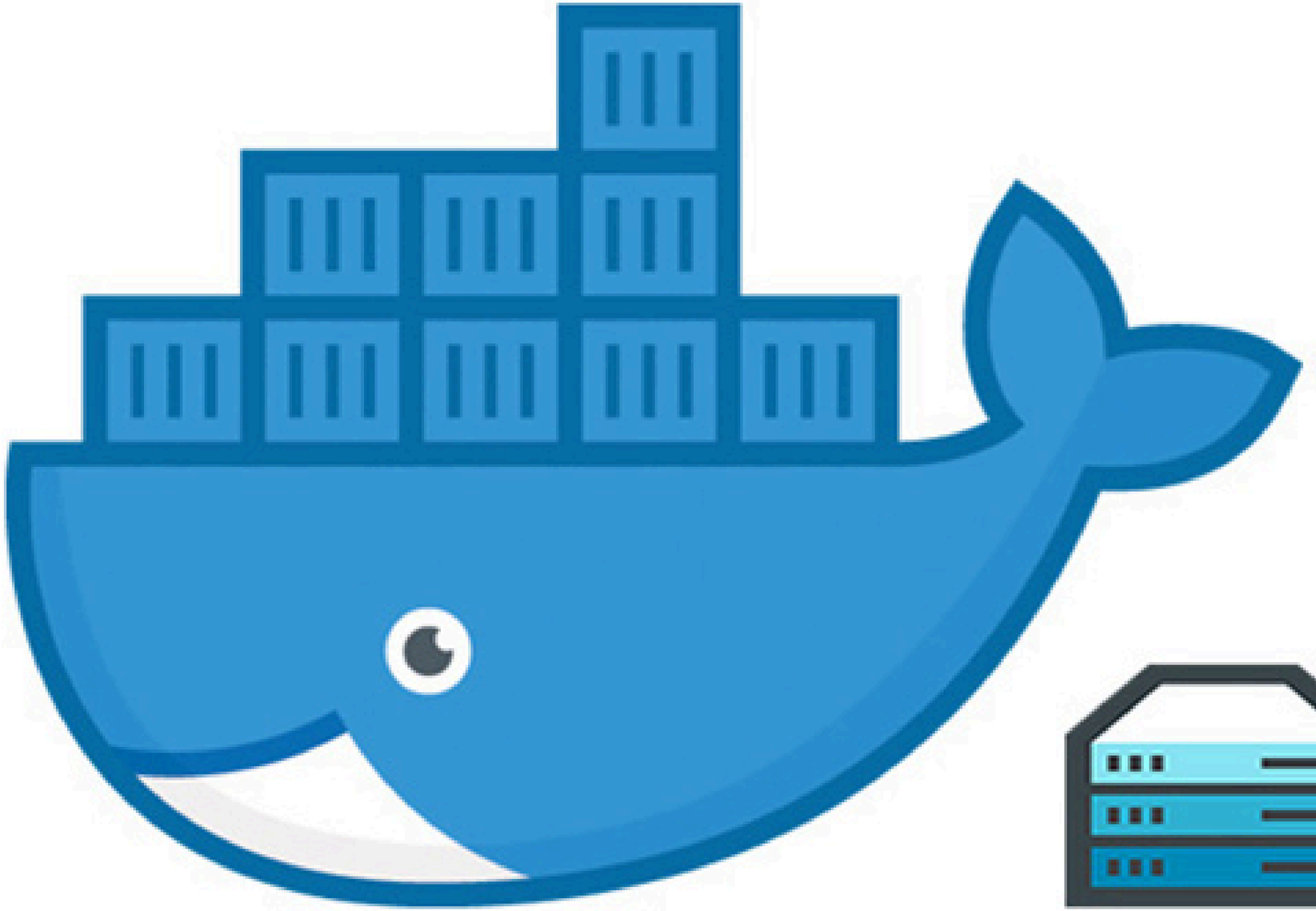
```
→  api git:(dev) ✗ docker run --name graphos-db -e POSTGRES_PASSWORD=12345 -d postgres
```

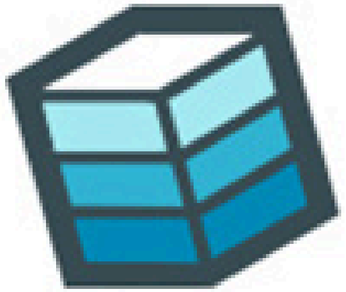
```
→  api git:(dev) ✗ docker run --name duque-db -e MYSQL_ROOT_PASSWORD=12345 -d mysql:5.7
```

```
→  api git:(dev) ✗ docker run --name schedup-mem-db -d redis
```

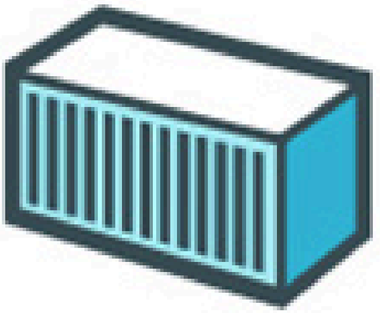
```
→  api git:(dev) ✗ docker run --name some-mongo -d mongo:tag
```

Diferentes serviços executando a partir de uma linha de código



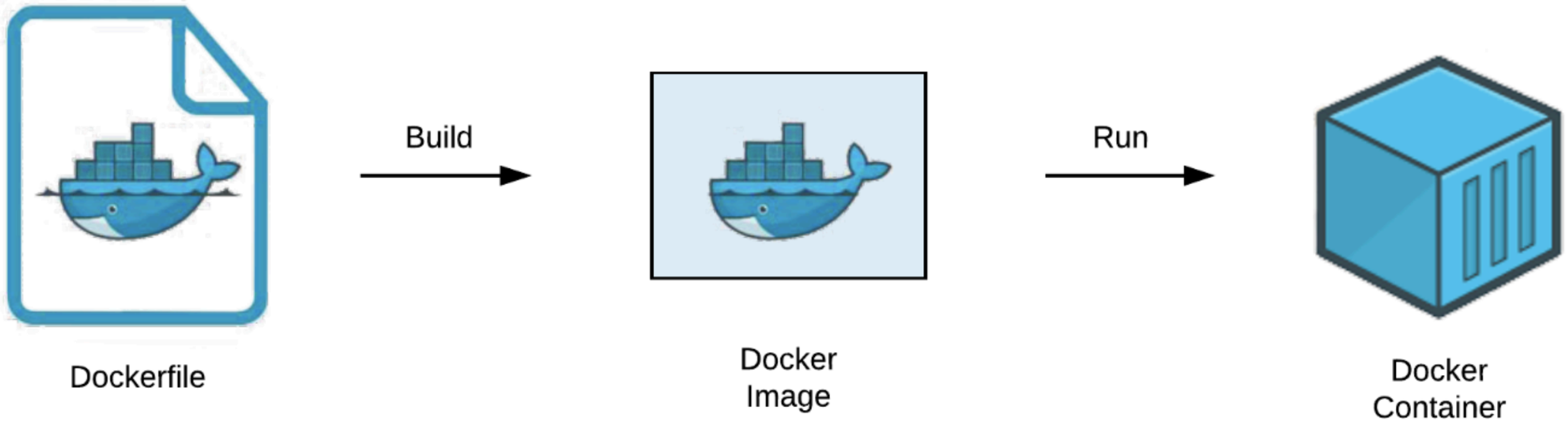


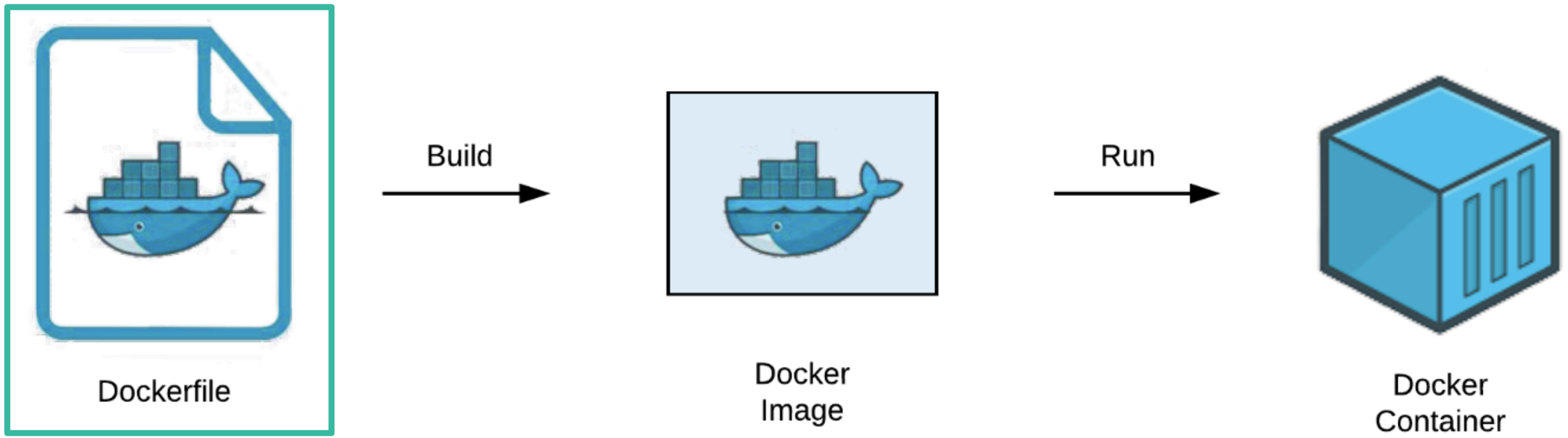
Build

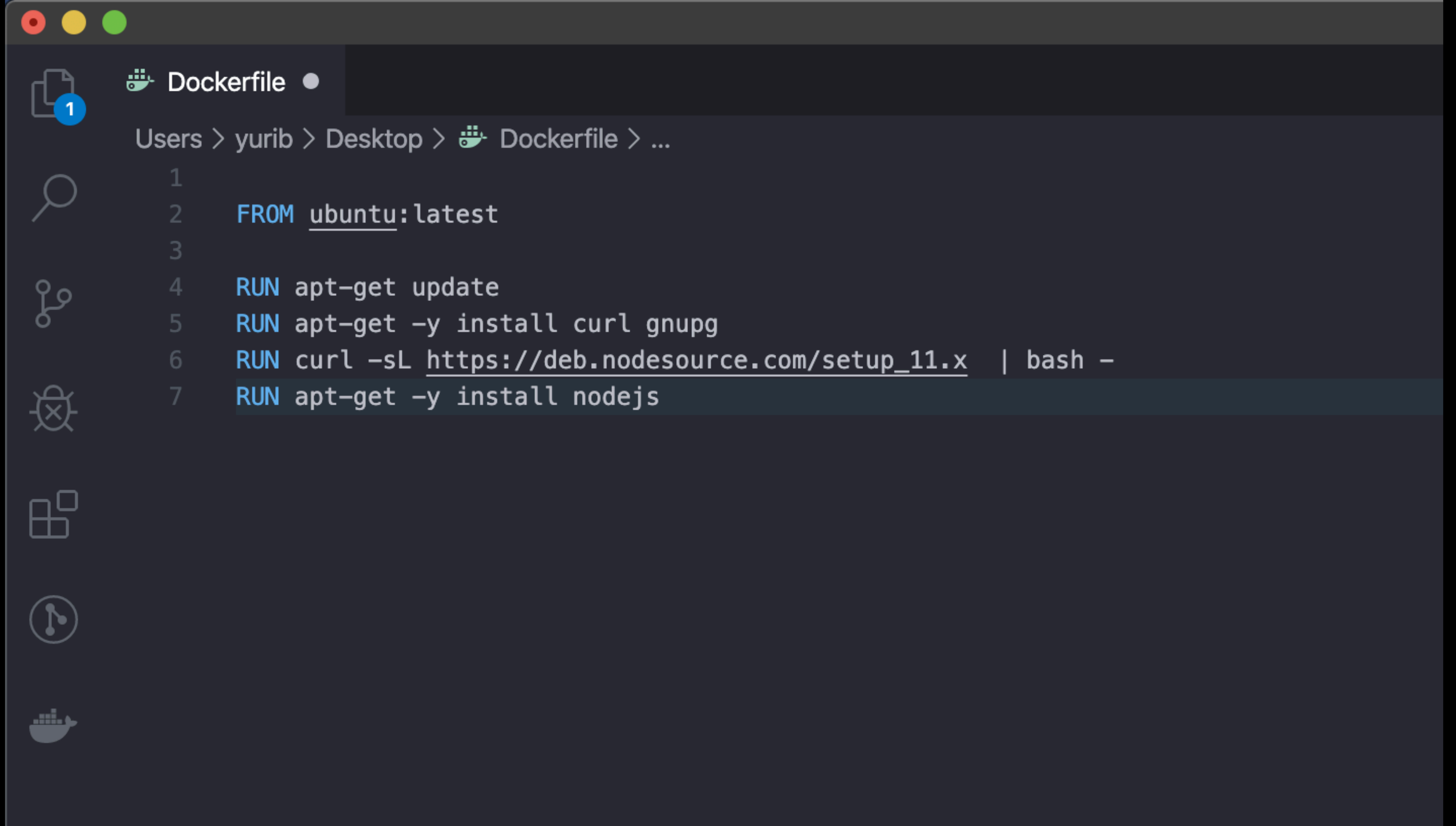


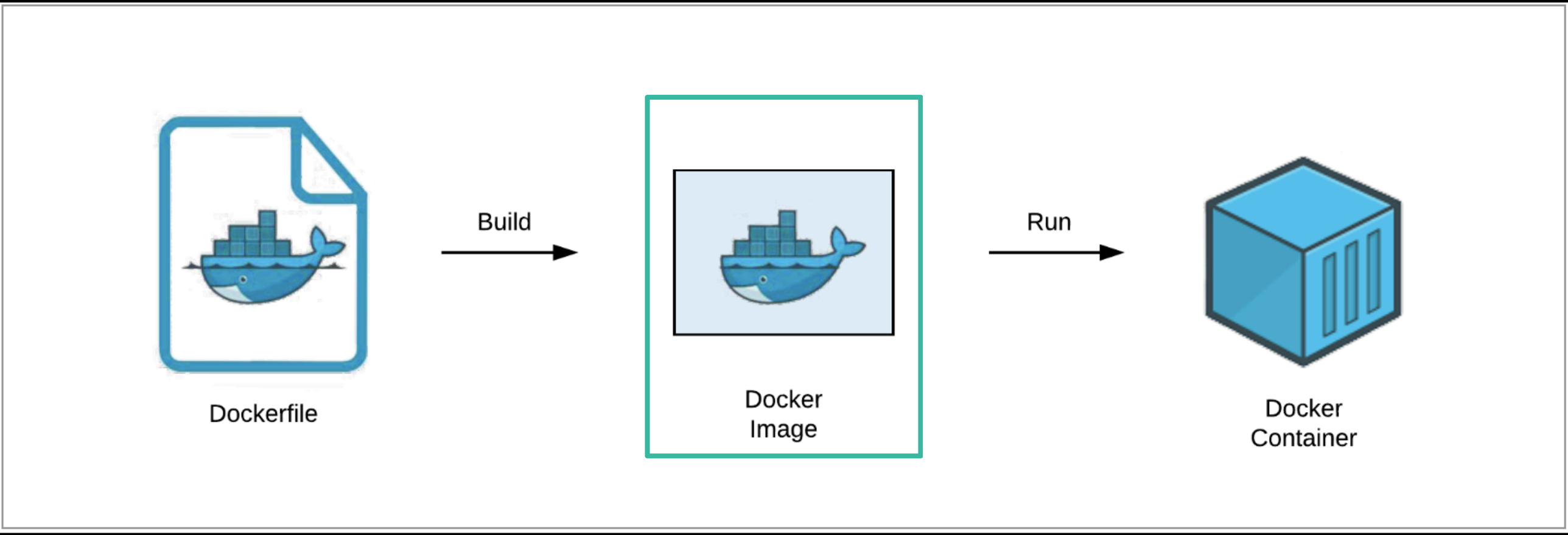
Ship

Run


Build
Run
Dockerfile
Docker
Image
Docker
Container


Dockerfile
Build
Docker
Image
Run
Docker
Container

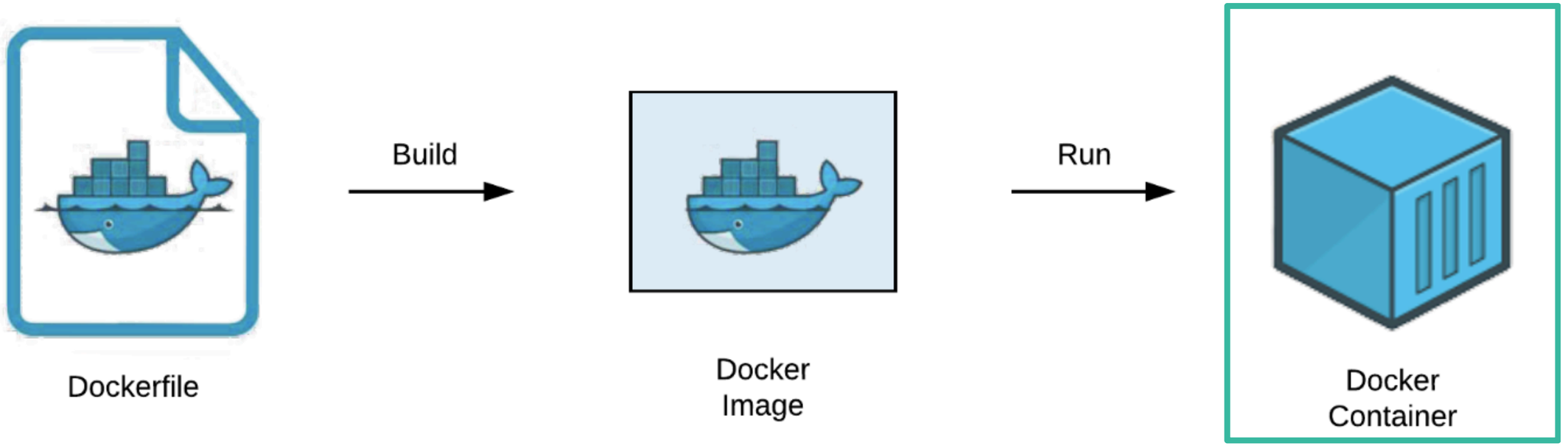

Dockerfile
Users > yurib > Desktop > Dockerfile > ...
1
FROM ubuntu:latest
3
RUN apt-get update
RUN apt-get -y install curl gnupg
RUN curl -sL https://deb.nodesource.com/setup_11.x | bash -
RUN apt-get -y install nodejs


Dockerfile
Build
Docker
Image
Run
Docker
Container

```
➜  Desktop docker build -t node_test .
```

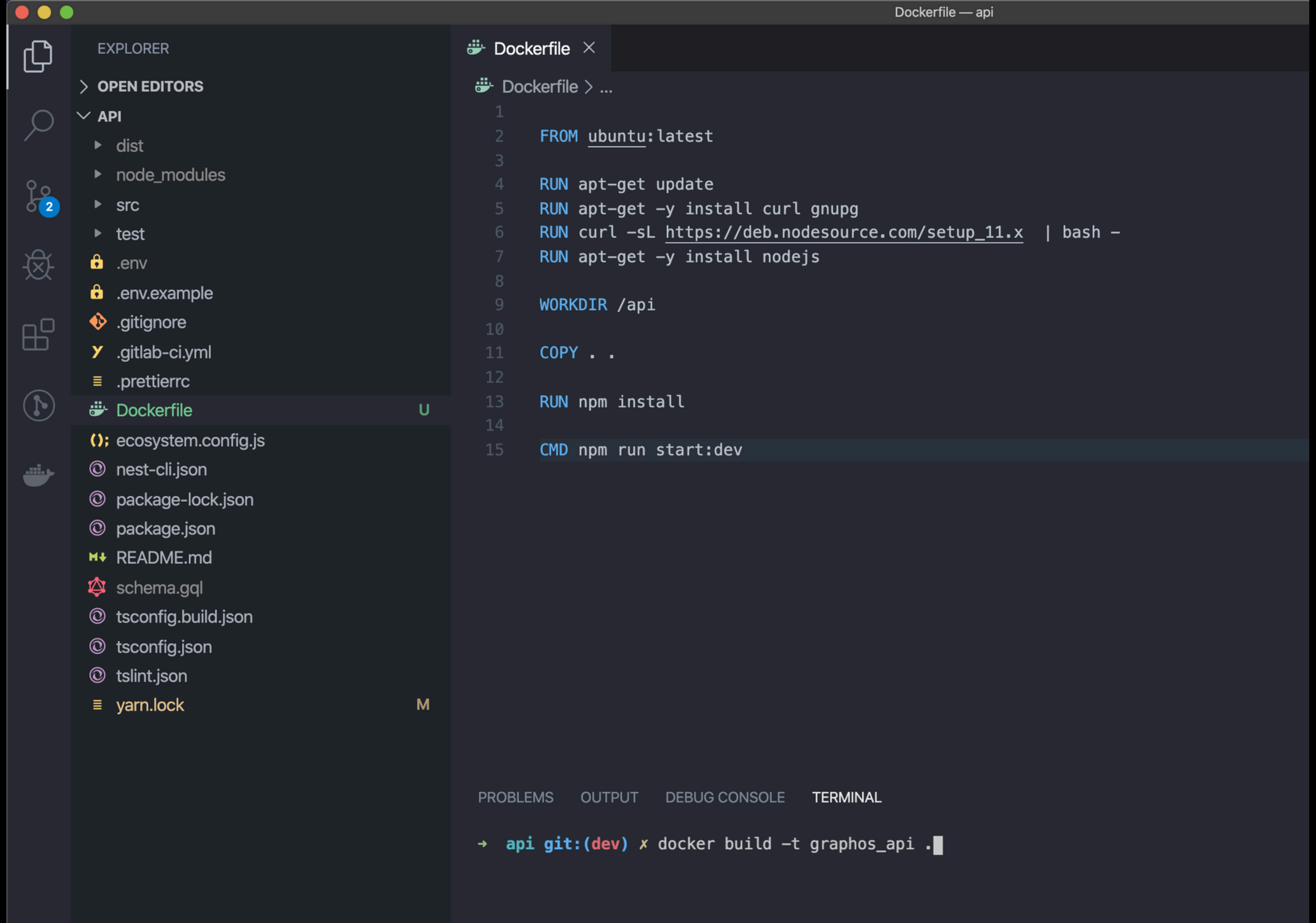

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
→ Desktop docker build -t node_test .
Sending build context to Docker daemon 21.97MB
Step 1/5 : FROM ubuntu:latest
latest: Pulling from library/ubuntu
2746a4a261c9: Pull complete
4c1d20cdee96: Pull complete
0d3160e1d0de: Pull complete
c8e37668deea: Pull complete
Digest: sha256:250cc6f3f3ffc5cdaa9d8f4946ac79821aafb4d3afc93928f0de9336eba21aa4
Status: Downloaded newer image for ubuntu:latest
---> 549b9b86cb8d
Step 2/5 : RUN apt-get update
---> Running in efe9f8ca15c0
Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic InRelease [242 kB]
Get:3 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [795 kB]
Get:4 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:6 http://archive.ubuntu.com/ubuntu bionic/main amd64 Packages [1344 kB]
Get:7 http://archive.ubuntu.com/ubuntu bionic/restricted amd64 Packages [13.5 kB]
Get:8 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Packages [19.2 kB]
Get:9 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [761 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic/multiverse amd64 Packages [186 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic/universe amd64 Packages [11.3 MB]
Get:12 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64 Packages [6781 B]
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [1057 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages [10.5 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages [32.7 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [1322 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic-backports/universe amd64 Packages [4244 B]
Get:18 http://archive.ubuntu.com/ubuntu bionic-backports/main amd64 Packages [2496 B]
Fetched 17.4 MB in 9s (1952 kB/s)
Reading package lists...
Removing intermediate container efe9f8ca15c0
---> dba82c5d54f5
Step 3/5 : RUN apt-get -y install curl gnupg
---> Running in 7842ce79467e
Reading package lists...
Building dependency tree...
Reading state information...
The following additional packages will be installed:
ca-certificates dirmngr gnupg-l10n gnupg-utils gpg gpg-agent gpg-wks-client
gpg-wks-server gpgconf gpgsm krb5-locales libasn1-8-heimdal libassuan0
libcurl4 libgssapi-krb5-2 libgssapi3-heimdal libhcrypto4-heimdal
libheimbase1-heimdal libheimntlm0-heimdal libhx509-5-heimdal libk5crypto3
libkeyutils1 libkrb5-26-heimdal libkrb5-3 libkrb5support0 libksba8
libldap-2.4-2 libldap-common libnghttp2-14 libnpth0 libpsl5 libreadline7
libroken18-heimdal librtmp1 libsasl2-2 libsasl2-modules libsasl2-modules-db
libsqlite3-0 libssl1.1 libwind0-heimdal openssl pinentry-curses publicsuffix
readline-common
Suggested packages:
dbus-user-session libpam-systemd pinentry-gnome3 tor parcimonie xloadimage
scdaemon krb5-doc krb5-user libsasl2-modules-gssapi-mit
| libsasl2-modules-gssapi-heimdal libsasl2-modules-ldap libsasl2-modules-otp
⊗ 0 ⚠ 0

```
➜  Desktop docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
node_test           latest              e7b298c2f459        26 minutes ago      225MB
ubuntu              latest              549b9b86cb8d        38 hours ago        64.2MB
```


Build
Run
Dockerfile
Docker
Image
Docker
Container

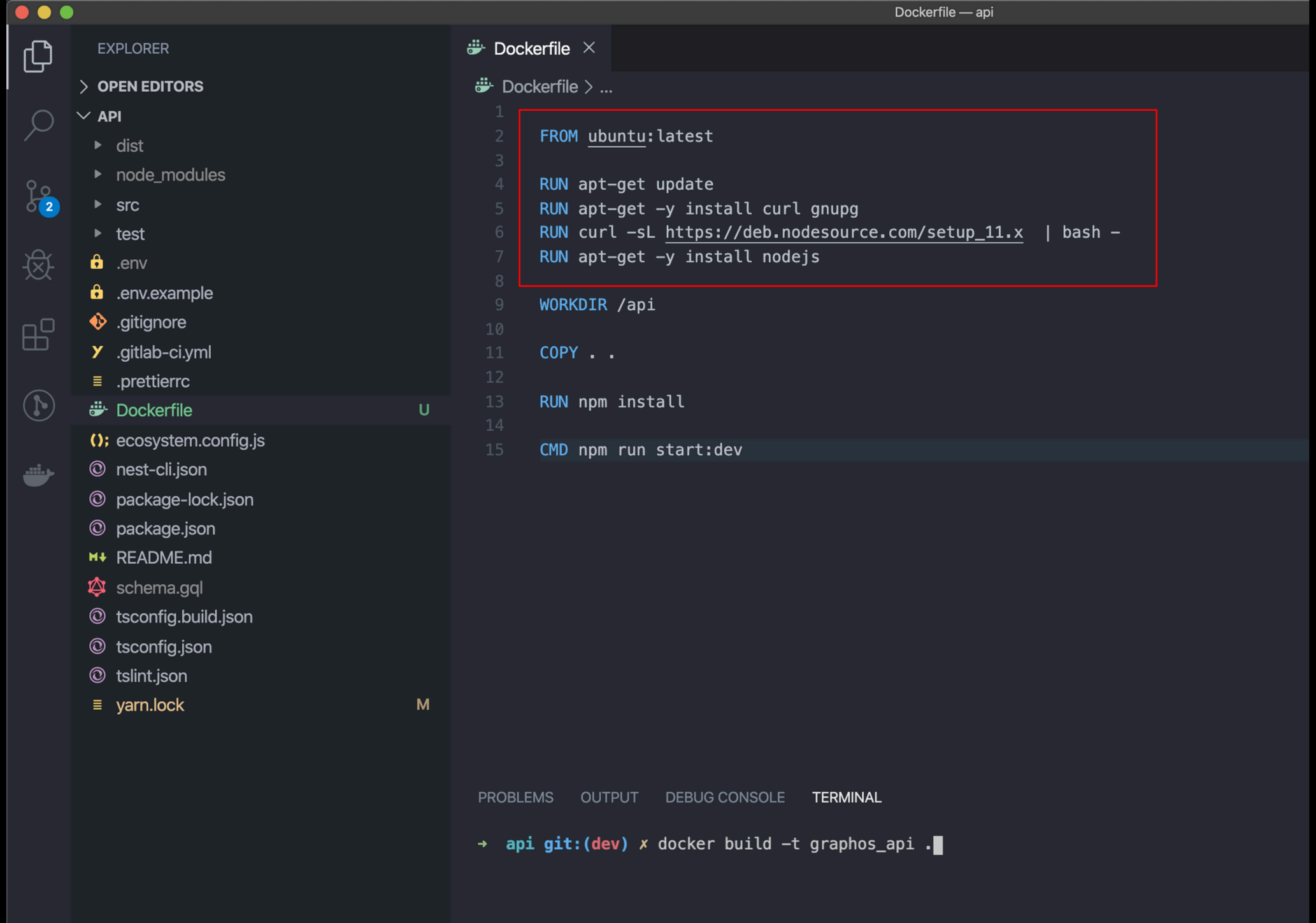
```
➜  Desktop docker run node_test
```

```
➜  api git:(dev) ✗ docker run node_test node -v
v11.15.0
➜  api git:(dev) ✗ docker run node_test npm -v
6.7.0
➜  api git:(dev) ✗ docker run node_test ls
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
➜  api git:(dev) ✗
```


Dockerfile — api
EXPLORER
OPEN EDITORS
API
dist
node_modules
src
test
.env
.env.example
.gitignore
.gitlab-ci.yml
.prettierrc
Dockerfile U
ecosystem.config.js
nest-cli.json
package-lock.json
package.json
README.md
schema.gql
tsconfig.build.json
tsconfig.json
tslint.json
yarn.lock M
Dockerfile
Dockerfile > ...
1
2 FROM ubuntu:latest
3
4 RUN apt-get update
5 RUN apt-get -y install curl gnupg
6 RUN curl -sL https://deb.nodesource.com/setup_11.x | bash -
7 RUN apt-get -y install nodejs
8
9 WORKDIR /api
10
11 COPY . .
12
13 RUN npm install
14
15 CMD npm run start:dev
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
→ api git:(dev) ✗ docker build -t graphos_api .

```
➜  api git:(dev) ✗ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
graphos_api         latest              4f9dd2b27172        9 minutes ago       432MB
ubuntu              latest              549b9b86cb8d        2 weeks ago         64.2MB
node                12-alpine           e12611f14560        2 weeks ago         84.9MB
```

```
➜  api git:(dev) ✗ docker run -p 3000:3000 graphos_api
```


Dockerfile — api
EXPLORER
OPEN EDITORS
API
dist
node_modules
src
test
.env
.env.example
.gitignore
.gitlab-ci.yml
.prettierrc
Dockerfile
U
ecosystem.config.js
nest-cli.json
package-lock.json
package.json
README.md
schema.gql
tsconfig.build.json
tsconfig.json
tslint.json
yarn.lock
M
Dockerfile
Dockerfile > ...
1 FROM ubuntu:latest
2
3 RUN apt-get update
4 RUN apt-get -y install curl gnupg
5 RUN curl -sL https://deb.nodesource.com/setup_11.x | bash -
6 RUN apt-get -y install nodejs
7
8 WORKDIR /api
9
10 COPY . .
11
12 RUN npm install
13
14 CMD npm run start:dev
15
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
→ api git:(dev) ✗ docker build -t graphos_api .

Access the world's largest library of container images

Official Images

couchbase
Official
10M+

postgres
Official
10M+

traefik
Official
10M+

httpd
Official
10M+

ubuntu
Official
10M+

mariadb
Official
10M+

busybox
Official
10M+

memcached
Official
10M+

mysql
Official
10M+

docker
Official
10M+

See all Official Images >

Filters

1 - 25 of 2,985,099 available images.

Most Popular

Docker Certified

- [ ] Docker Certified

Images

- [ ] **Verified Publisher** *Docker Certified And Verified Publisher Content*
- [ ] **Official Images** *Official Images Published By Docker*

Categories

- [ ] Analytics
- [ ] Application Frameworks
- [ ] Application Infrastructure
- [ ] Application Services
- [ ] Base Images
- [ ] Databases
- [ ] DevOps Tools
- [ ] Featured Images
- [ ] Messaging Services
- [ ] Monitoring
- [ ] Operating Systems
- [ ] Programming Languages
- [ ] Security
- [ ] Storage

VERIFIED PUBLISHER

MySQL Server Enterprise Edition DOCKER CERTIFIED

By Oracle • Updated a year ago

The world's most popular open source database system

Container | Docker Certified | Linux | x86-64 | Databases

OFFICIAL IMAGE

couchbase

Updated 37 minutes ago

10M+ Downloads | 521 Stars

Couchbase Server is a NoSQL document database with a distributed architecture.

Container | Linux | x86-64 | Storage | Application Frameworks

OFFICIAL IMAGE

ubuntu

Updated 39 minutes ago

10M+ Downloads | 10K+ Stars

Ubuntu is a Debian-based Linux operating system based on free software.

Container | Linux | x86-64 | 386 | PowerPC 64 LE | ARM | IBM Z | ARM 64 | Base Images | Operating Systems

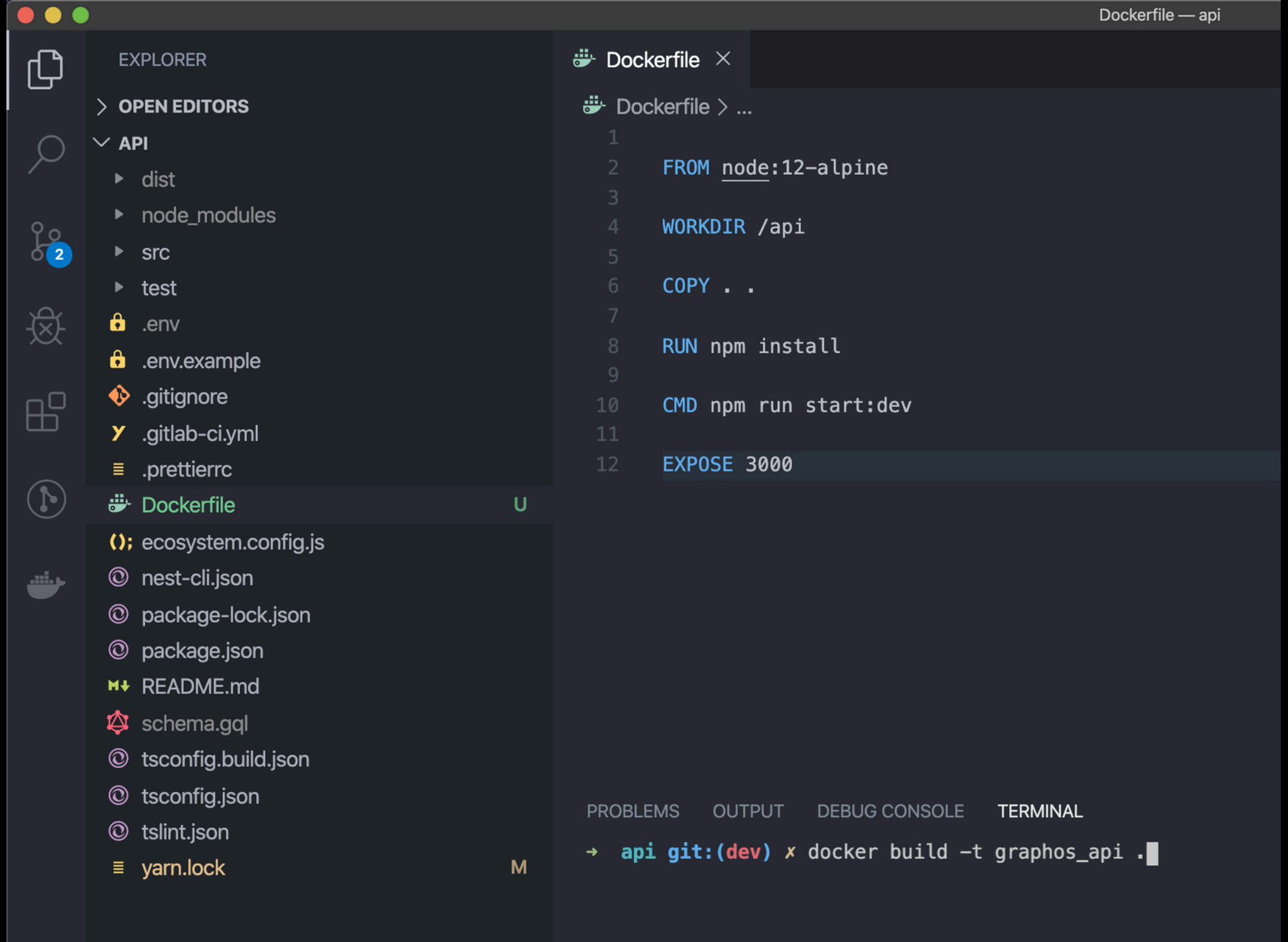
OFFICIAL IMAGE

redis

Updated 39 minutes ago

10M+ Downloads | 7.7K Stars

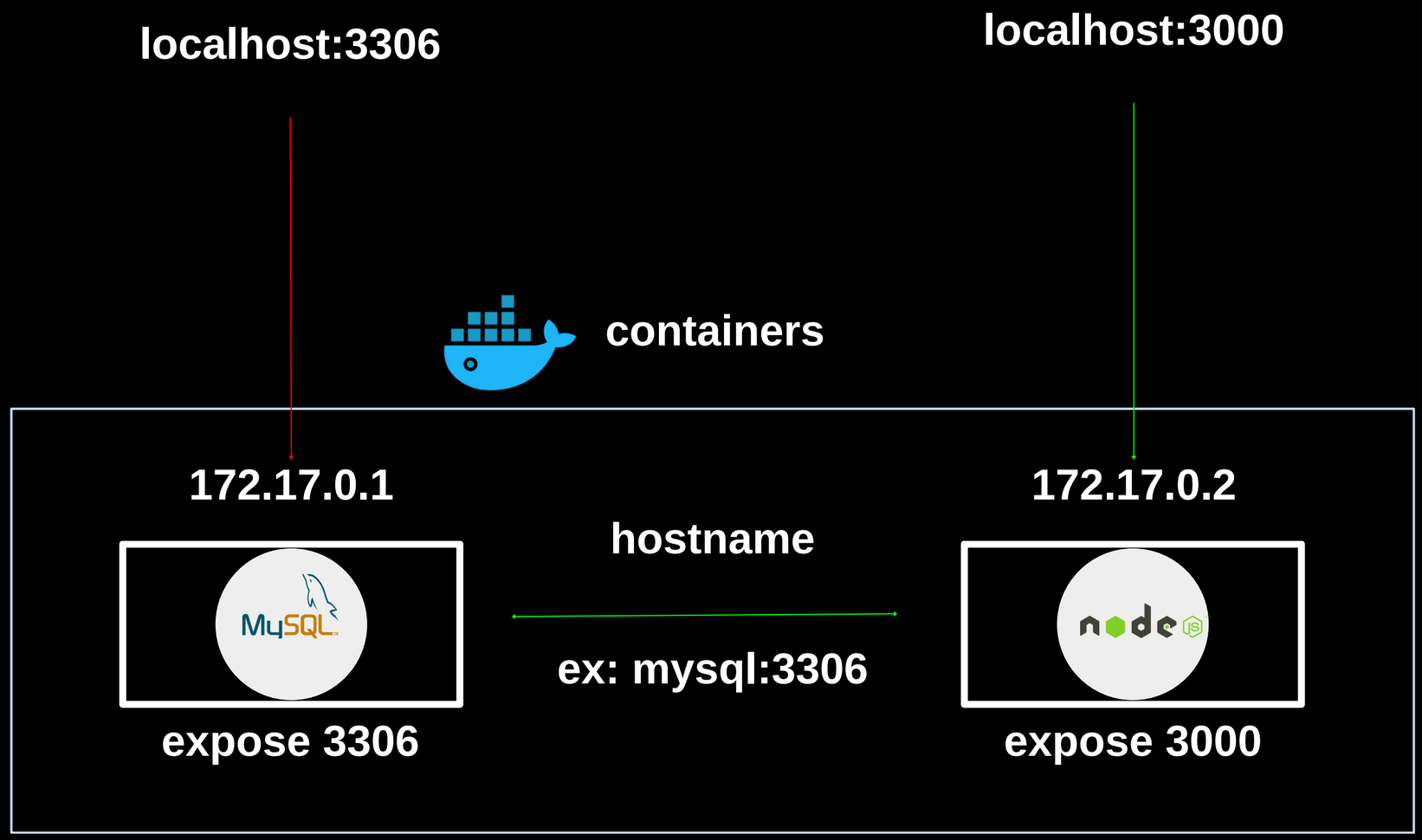
Redis is an open source key-value store that functions as a data structure server.


Dockerfile — api
EXPLORER
OPEN EDITORS
API
dist
node_modules
src
test
.env
.env.example
.gitignore
.gitlab-ci.yml
.prettierrc
Dockerfile U
ecosystem.config.js
nest-cli.json
package-lock.json
package.json
README.md
schema.gql
tsconfig.build.json
tsconfig.json
tslint.json
yarn.lock M
Dockerfile
Dockerfile > ...
1
2 FROM node:12-alpine
3
4 WORKDIR /api
5
6 COPY . .
7
8 RUN npm install
9
10 CMD npm run start:dev
11
12 EXPOSE 3000
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
→ api git:(dev) ✗ docker build -t graphos_api .

```
➜  api git:(dev) ✗ docker run -p 3000:3000 graphos_api
```

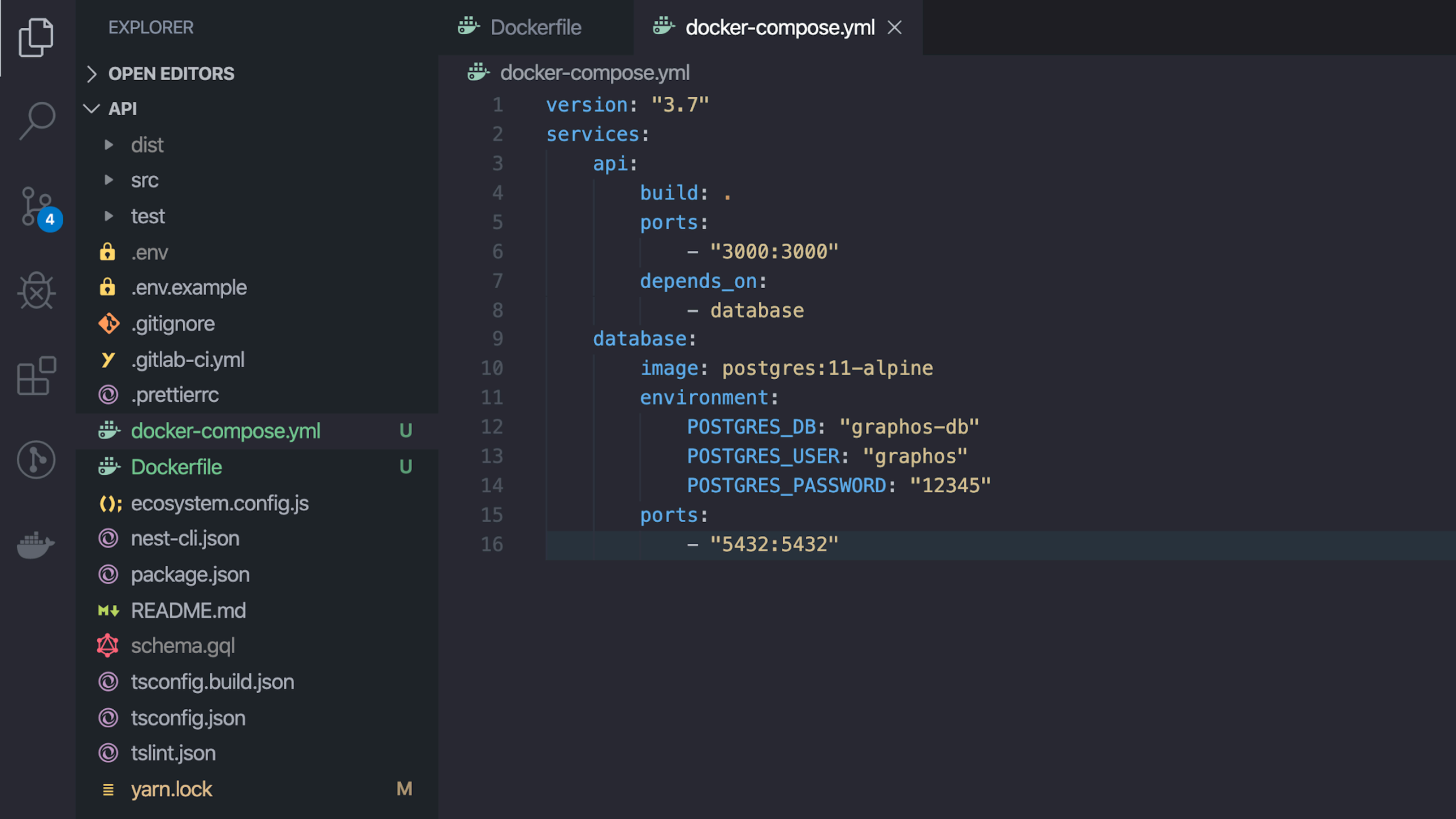
```
7:49:09 PM - Starting compilation in watch mode...


7:49:30 PM - Found 0 errors. Watching for file changes.
[Nest] 28   - 12/20/2019, 7:49:31 PM   [NestFactory] Starting Nest application...
[Nest] 28   - 12/20/2019, 7:49:31 PM   [InstanceLoader] AppModule dependencies initialized +60ms
[Nest] 28   - 12/20/2019, 7:49:31 PM   [InstanceLoader] ConfigModule dependencies initialized +1ms
[Nest] 28   - 12/20/2019, 7:49:31 PM   [InstanceLoader] TypeOrmModule dependencies initialized +1ms
[Nest] 28   - 12/20/2019, 7:49:31 PM   [InstanceLoader] PassportModule dependencies initialized +1ms
[Nest] 28   - 12/20/2019, 7:49:31 PM   [InstanceLoader] SharedModule dependencies initialized +1ms
[Nest] 28   - 12/20/2019, 7:49:31 PM   [InstanceLoader] JwtModule dependencies initialized +1ms
[Nest] 28   - 12/20/2019, 7:49:31 PM   [InstanceLoader] GraphQLModule dependencies initialized +28ms
[Nest] 28   - 12/20/2019, 7:49:31 PM   [TypeOrmModule] Unable to connect to the database. Retrying (1)... +4ms
Error: connect ECONNREFUSED 127.0.0.1:32800
    at TCPConnectWrap.afterConnect [as oncomplete] (net.js:1129:14)
```

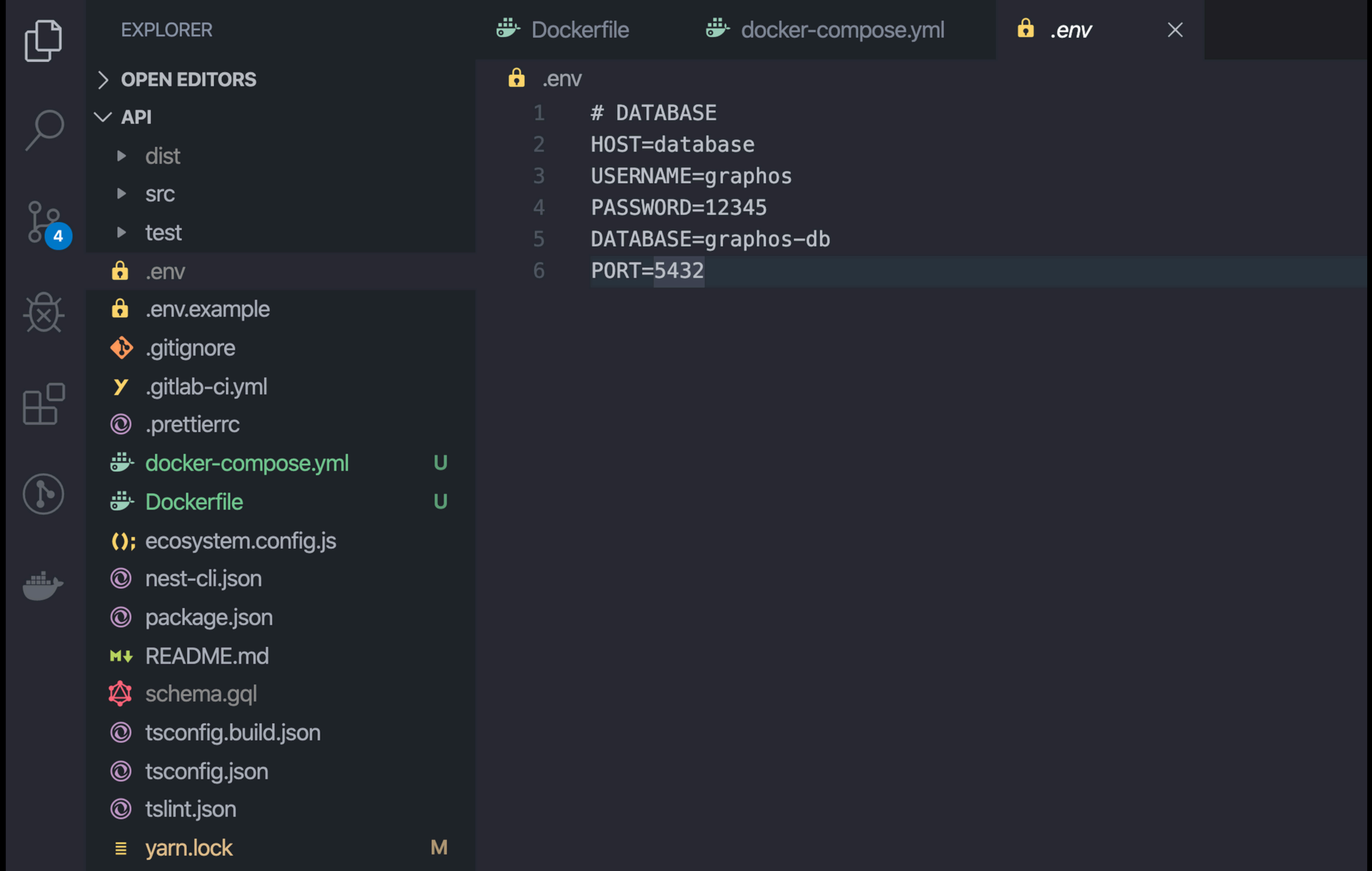
```
➜  api git:(dev) ✗ docker run -p 3000:3000 graphos_api

➜  api git:(dev) ✗ docker run mysql
```



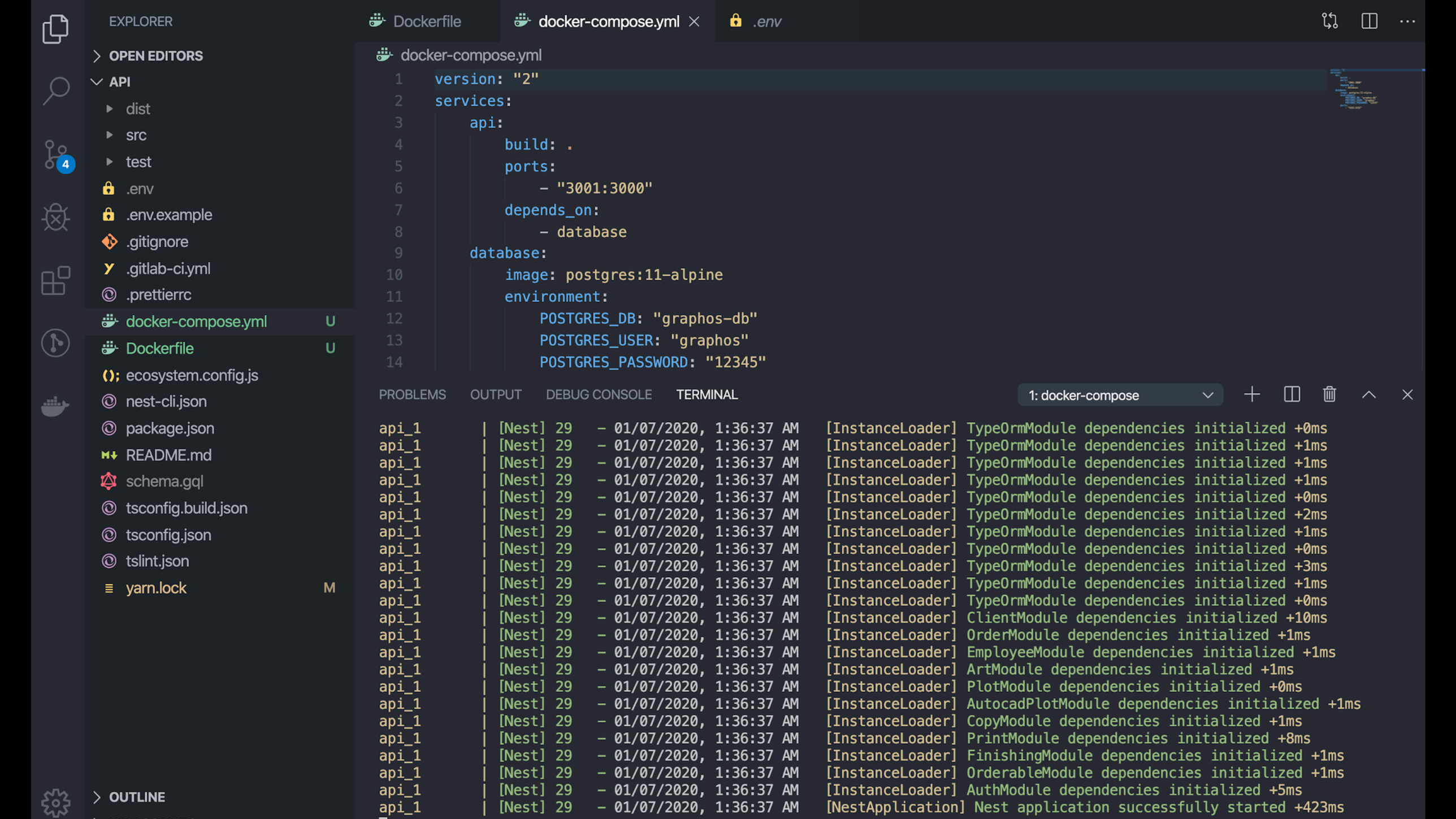
Docker

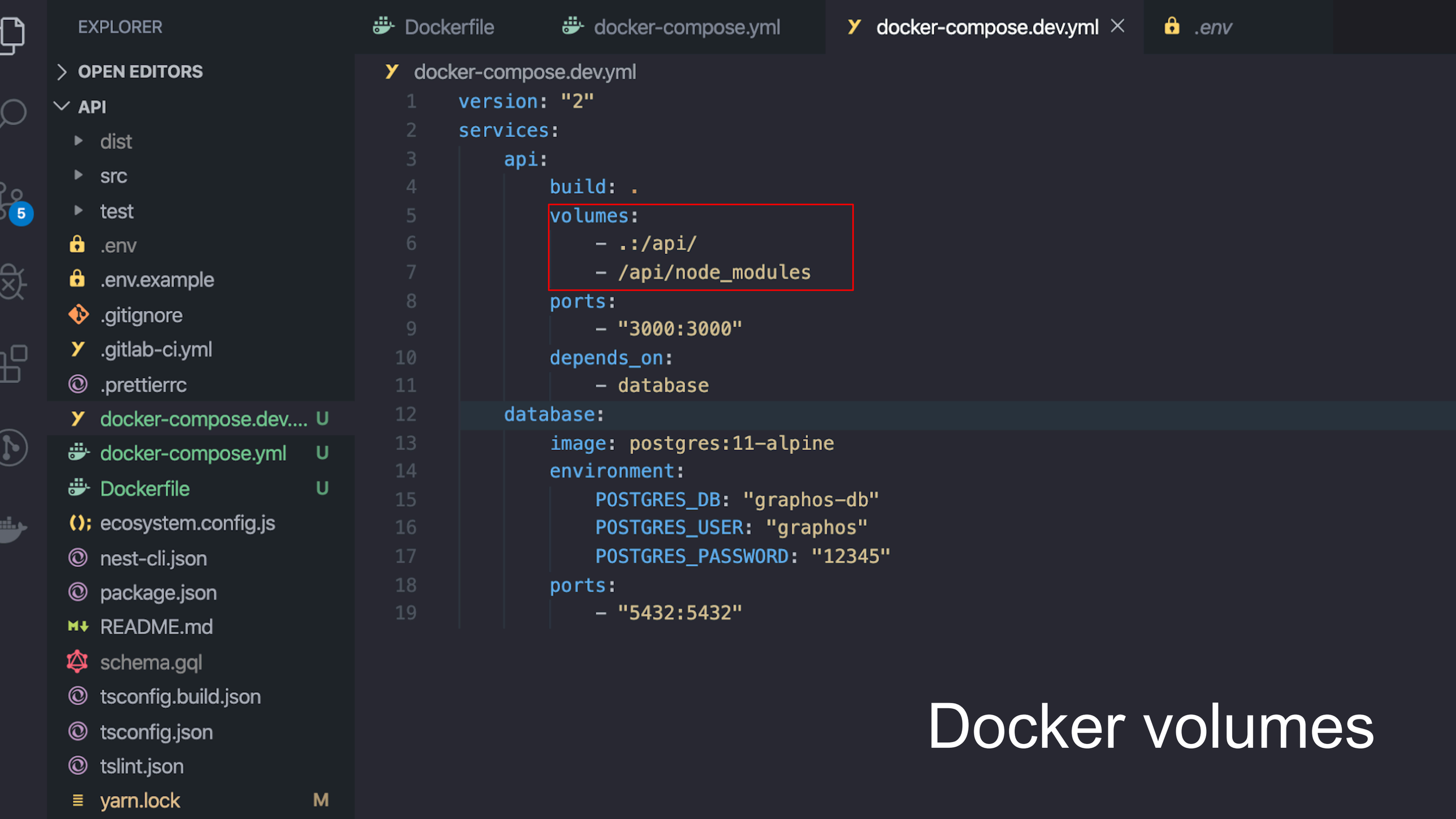
Compose


EXPLORER
Dockerfile
docker-compose.yml
OPEN EDITORS
API
dist
src
test
.env
.env.example
.gitignore
.gitlab-ci.yml
.prettierrc
docker-compose.yml U
Dockerfile U
ecosystem.config.js
nest-cli.json
package.json
README.md
schema.gql
tsconfig.build.json
tsconfig.json
tslint.json
yarn.lock M
docker-compose.yml
1 version: "3.7"
2 services:
3 api:
4 build: .
5 ports:
6 - "3000:3000"
7 depends_on:
8 - database
9 database:
10 image: postgres:11-alpine
11 environment:
12 POSTGRES_DB: "graphos-db"
13 POSTGRES_USER: "graphos"
14 POSTGRES_PASSWORD: "12345"
15 ports:
16 - "5432:5432"


EXPLORER
Dockerfile
docker-compose.yml
.env
OPEN EDITORS
API
dist
src
test
.env
.env.example
.gitignore
.gitlab-ci.yml
.prettierrc
docker-compose.yml U
Dockerfile U
ecosystem.config.js
nest-cli.json
package.json
README.md
schema.gql
tsconfig.build.json
tsconfig.json
tslint.json
yarn.lock M
.env
1 # DATABASE
2 HOST=database
3 USERNAME=graphos
4 PASSWORD=12345
5 DATABASE=graphos-db
6 PORT=5432

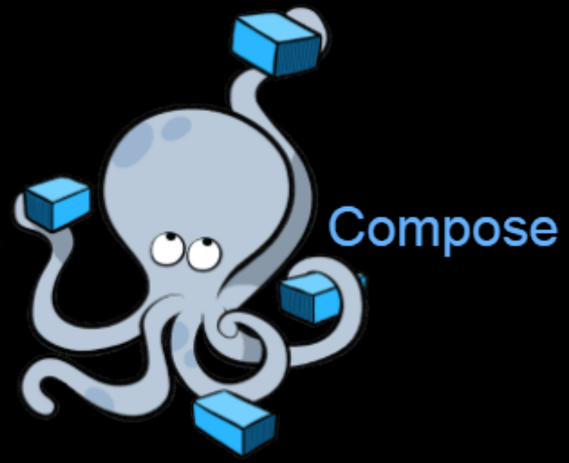
```
➜  api git:(dev) ✗ docker-compose up
```


EXPLORER
Dockerfile
docker-compose.yml
.env
OPEN EDITORS
API
dist
src
test
.env
.env.example
.gitignore
.gitlab-ci.yml
.prettierrc
docker-compose.yml U
Dockerfile U
ecosystem.config.js
nest-cli.json
package.json
README.md
schema.gql
tsconfig.build.json
tsconfig.json
tslint.json
yarn.lock M
OUTLINE
docker-compose.yml
version: "2"
services:
api:
build: .
ports:
- "3001:3000"
depends_on:
- database
database:
image: postgres:11-alpine
environment:
POSTGRES_DB: "graphos-db"
POSTGRES_USER: "graphos"
POSTGRES_PASSWORD: "12345"
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: docker-compose
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +0ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +0ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +2ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +0ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +3ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] TypeOrmModule dependencies initialized +0ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] ClientModule dependencies initialized +10ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] OrderModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] EmployeeModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] ArtModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] PlotModule dependencies initialized +0ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] AutocadPlotModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] CopyModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] PrintModule dependencies initialized +8ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] FinishingModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] OrderableModule dependencies initialized +1ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [InstanceLoader] AuthModule dependencies initialized +5ms
api_1 | [Nest] 29 - 01/07/2020, 1:36:37 AM [NestApplication] Nest application successfully started +423ms


EXPLORER
Dockerfile
docker-compose.yml
docker-compose.dev.yml
.env
OPEN EDITORS
API
dist
src
test
.env
.env.example
.gitignore
.gitlab-ci.yml
.prettierrc
docker-compose.dev.... U
docker-compose.yml U
Dockerfile U
ecosystem.config.js
nest-cli.json
package.json
README.md
schema.gql
tsconfig.build.json
tsconfig.json
tslint.json
yarn.lock M
docker-compose.dev.yml
version: "2"
services:
api:
build: .
volumes:
- .:/api/
- /api/node_modules
ports:
- "3000:3000"
depends_on:
- database
database:
image: postgres:11-alpine
environment:
POSTGRES_DB: "graphos-db"
POSTGRES_USER: "graphos"
POSTGRES_PASSWORD: "12345"
ports:
- "5432:5432"
Docker volumes

Mais comandos

- docker-compose up
- docker-compose stop
- docker-compose restart
- docker-compose down
- docker-compose build
- docker-compose logs

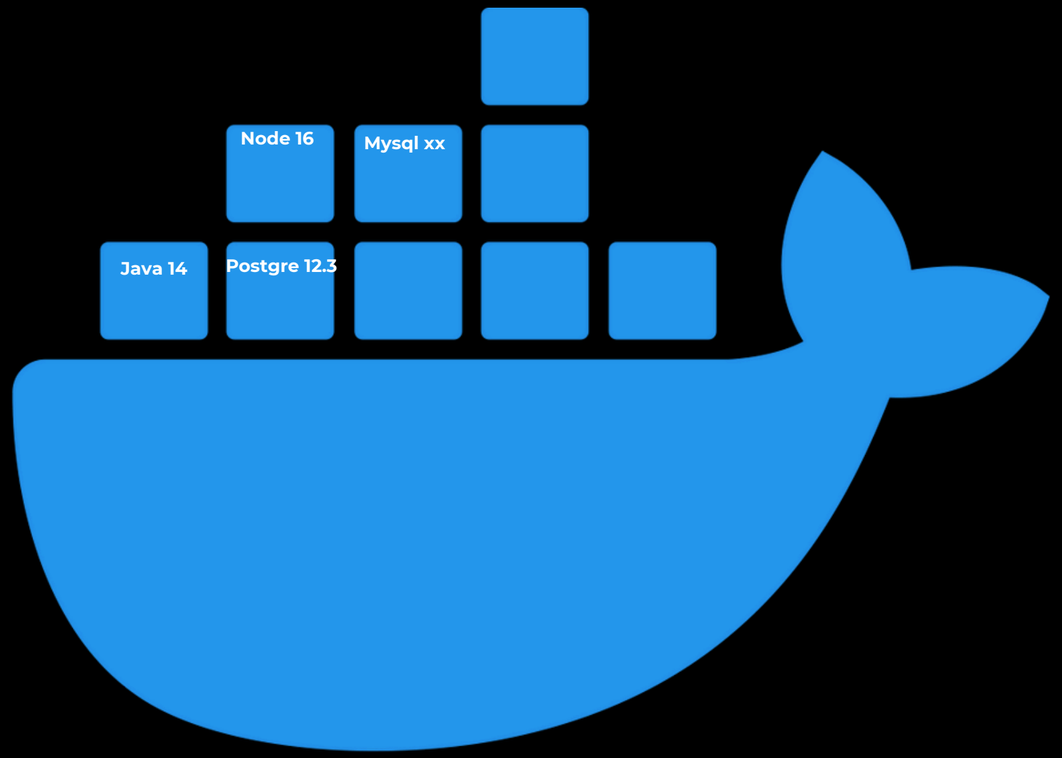


```
➜  api git:(dev) ✗ docker ps
CONTAINER ID        IMAGE                COMMAND                  CREATED             STATUS              PORTS                    NAMES
a35fc7f2683c        api_api              "docker-entrypoint.s…"   28 minutes ago      Up 31 seconds       0.0.0.0:3001->3000/tcp   api_api_1
b0ae9f6bb5cf        postgres:11-alpine   "docker-entrypoint.s…"   28 minutes ago      Up 32 seconds       0.0.0.0:5432->5432/tcp   api_database_1
```

```
➜  api git:(dev) ✗ docker exec -it api_database_1 bash
bash-5.0# psql -d graphos-db -U graphos
psql (11.6)
Type "help" for help.

graphos-db=#
```

```
➜  api git:(dev) ✗ docker stats
```

```
CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT     MEM %
a35fc7f2683c        api_api_1           5.35%               401.1MiB / 1.952GiB   20.06%
b0ae9f6bb5cf        api_database_1      0.00%               4.199MiB / 1.952GiB   0.21%
```



docker®